DERWENT-ACC-NO: 1997-158602

DERWENT-WEEK: 199715

TITLE: Batch job execution management for automatic operation at night - involves checking premise conditions such as execution schedule judging and execution result state after executing job

PATENT-ASSIGNEE: NEC CORP[NIDE]

PRIORITY-DATA: 1995JP-0085352 (April 11, 1995)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
JP 08286778 A	November 1, 1996	N/A	008	G06F 001/00

APPLICATION-DATA:

PUB-NO	APPL-DESCRIPTOR	APPL-NO	APPL-DATE
JP 08286778A	N/A	1995JP-0085352	April 11, 1995

INT-CL (IPC): G06F001/00, G06F011/34

ABSTRACTED-PUB-NO: JP 08286778A

BASIC-ABSTRACT:

The batch processing method is used for batch processing containing continuous jobs in a job stream which is to be performed automatically. A job execution management file comprising a job schedule record, job check record and job result renewal record, used.

Each execution propriety judging of job and execution result

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平8－286778

(43)公開日　平成8年(1996)11月1日

(51)Int.Cl.*	識別記号	庁内整理番号	FI		技術表示箇所
G06F 1/00	370		G06F 1/00	370A	
11/34		7313－5B	11/34	A	

審査請求 有　請求項の数5 OL （全 8 頁）

(21)出願番号　特願平7－85352

(22)出願日　平成7年(1995)4月11日

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目7番1号

(72)発明者　長谷川　正裕
東京都港区芝五丁目7番1号　日本電気株式会社内

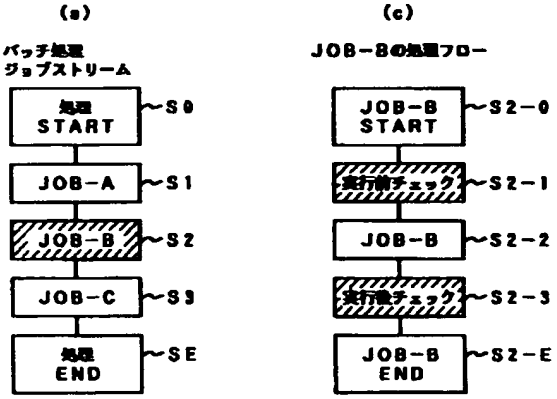
(74)代理人　弁理士　後藤　洋介　（外2名）

(54)【発明の名称】　ジョブ実行管理方法及び装置

(57)【要約】

【目的】　連続するバッチ形式のトランザクション処理を自動的に実行する業務において，各々のジョブ（ＪＯＢ）の実行スケジュールを含む実行可否判定，および各ＪＯＢの実行結果管理を行う。

【構成】　連続するＪＯＢから成るバッチ処理をジョブストリームによって自動的に実行する際，ＪＯＢスケジュールレコード，ＪＯＢチェックレコード，ＪＯＢ結果更新レコードから構成されるＪＯＢ実行管理ファイルを用いて，実行前チェックおよび実行後チェックを行うことによって，各々のＪＯＢの実行可否判定および実行結果管理を行う。なお，実行前チェックにおいては，実行スケジュール判定および，前提条件となるＪＯＢの実行結果状態判定を，また，実行後チェックにおいては，各々のＪＯＢを前提条件とするＪＯＢのＪＯＢチェックレコードに自ＪＯＢの実行結果状態を格納する。



(b)

各ＪＯＢの実行スケジュールと前提条件ＪＯＢ

JOB	スケジュール	前提条件ＪＯＢ
A	実行	－
B	実行	A
C	不実行	A，B

1

【特許請求の範囲】

【請求項1】　夜間等の無人自動運転を含む，複数のジョブからなるバッチ処理を行う汎用機システムにおいて，個々のジョブの実行スケジュールを実行するにあたって，前記個々のジョブの前提条件となるジョブを登録するとともに，その各々のジョブの実行結果を記録することによって，連続する前記複数のジョブの実行可否判定及びジョブの実行結果管理を自動的に行うことを特徴とするジョブ実行管理方法。

【請求項2】　請求項1記載のジョブ実行管理方法において，前記実行可否判定は，選択された個々のジョブの実行前に行われる実行予定チェックによりなされ，前記実行結果管理は前記選択された個々のジョブの実行後に行われる実行結果のチェックによりなされることを特徴とするジョブ実行管理方法。

【請求項3】　請求項2記載のジョブ実行管理方法において，前記実行予定チェックは，予め作成された各ジョブの時期記録及び実行可否記録を有するジョブスケジュールレコードと，前記前提条件となるジョブ登録において，各ジョブが，正常，異常，及び不実行の内のいずれかを示すジョブチェックレコードとを参照することによってなされ，前記実行結果のチェックは，ジョブ識別とチェックレコード逆引テーブルから構成されたジョブ結果更新レコードと前記ジョブチェックレコードとを参照して行われることを特徴とするジョブ実行管理方法。

【請求項4】　夜間等の無人自動運転を含む，複数のジョブからなるバッチ処理を行う汎用機システムにおいて，個々のジョブの実行スケジュールと実行するにあたって，前提条件となるジョブを登録するとともに，その各々のジョブの実行結果を記録するファイル作成手段と，作成されたファイルに基づいて連続するジョブの実行可否判定を行う実行前にチェックする手段とジョブの実行結果管理を自動的に行う実行結果チェック手段とを備えていることを特徴とするジョブ実行管理装置。

【請求項5】　請求項4記載のジョブ実行管理装置において，予め作成された実行時期データ及び各ジョブが実行されているか否かを記録したジョブスケジュールレコードと，ジョブを行う際に新規に作成された前提条件において各ジョブが正常，異常，及び不実行の内のいずれかの状態であるかを記録したジョブチェックレコードと，ジョブ識別とチェックレコード逆引きテーブルから構成されたジョブ結果更新レコードとを備えたジョブ実行管理ファイルを備えていることを特徴とするジョブ実行管理装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は，人為的に実施される様々なエントリ業務の結果発生したトランザクションデータをもとに実施するバッチ形式の複数ジョブ（以下，JOBと呼ぶ）の実行管理業務に関し，特に個々のJOB

2

の実行可否判定・実行結果管理を同一バッチ処理内で自動的に行うJOB実行管理方法及びそれを実施するための装置に関する。

【0002】

【従来の技術】従来，トランザクション処理を行うバッチ形式の複数JOBを夜間等に無人状態で行う際には，その連続するバッチ処理を行うようにコーディングされたジョブストリームをシステム上に組み込むことによって，各JOBの起動のみを自動的に行っていた。したがって，このシステムでは，各々のJOBの実施スケジュール管理を含む実行可否判定を行うこと，及びそのJOBの実行結果を管理して後続する関連JOBの実行可否判定基準とすることは不可能な状態であった。

【0003】

【発明が解決しようとする課題】上述したように，各JOBの起動情報のみを格納したジョブストリームを用いて連続するバッチ形式のトランザクション処理を自動的に実行するため，各々のJOBの実行スケジュール管理を含む実行可否判定，および各JOBの実行結果管理は不可能であり，実行にあたっての前提条件が揃わないままに，次なるJOBが実行されることも多い。また，実行結果確認も，そのバッチ処理完了後に人為的に行う以外には，不可能であった。したがって，処理異常を認識しないまま翌日のエントリ処理を開始してしまうか，もしくは認識したとしても該当バッチJOB実行前のファイル環境にセーブ（SAVE）媒体から戻してリラン（RERUN）する必要が生じ，リカバリ（RECOVERY）に多大な工数がかかるとともに，業務自体にも機能の提供が不可能になるなど，不祥事が生じるという問題点があった。

【0004】そこで，本発明の技術的課題は，作業の正確性の向上，作業軽減はもとより，システム異常の場合，リカバリ工数を削減し，ひいては業務自体に支障の少ないシステムサポートが実現可能となる自動運転形式のバッチ形式ジョブ実行管理方式を提供することにある。

【0005】

【課題を解決するための手段】本発明のバッチ形式JOBの自動実行管理システムにおいては，上述したような課題を解決するために，新規JOB作成時にその各々のJOBごとに前提となるJOB（処理）を作成するとともに，また，運用時にその実行スケジュールを登録しておくことによって，JOB実行管理ファイルを作成（更新）する機能，また，ジョブストリームに基づいてバッチ処理が実施される際に，前提条件となるJOBがすべて正常に完了しているかどうか，および当該JOBの実施スケジュールを前述のJOB実行管理ファイルにもとづいて確認し，実行可否を判定する実行前チェックし，且つ各々のJOB実行結果をJOB実行管理ファイルに格納するJOB実行後チェックを行う。

【0006】

【実施例】以下，本発明の実施例について，図面を参照して説明する。

【0007】図1（a）は本発明の実施例における構成と流れを示す説明図，図1（b）は図1の例における各JOBの実行スケジュールと前提条件JOBを示す図である。図1（c）は図1（a）のJOB－Bの具体的な処理フローを示す図である。

【0008】また，図2（a）はJOBスケジュールレコードを示す図，図2（b）はJOBチェックレコードを示す図，図2（c）はJOB結果更新レコードを示す図，図2（d）は各チェックの際，参照されるJOB実行管理ファイルを示す図である。

【0009】図1（a）には，JOB－A，JOB－B，JOB－Cの3つのJOBからなるバッチ処理のジョブストリームが示されている。図1（a）では，ステップS0において，処理がスタートし，ステップS1でJOB－A，ステップS2でJOB－B，次いで，ステップS3でJOB－Cが実行され，ステップSEで処理が終了となるジョブストリームが示されている。

【0010】図1（b）を参照すると，図1（a）におけるJOB－A，JOB－B，JOB－Cの各JOBに設定されている実行スケジュールと，各JOBを実行するにあたっての前提条件となるJOBが示されている。図1（b）から明らかなように，JOB－Aが実行されるときの前提条件が無く，また，JOB－Bは，実行される前提条件としてJOB－Aが実行された後に行われることを示している。また，JOB－Cは，JOB－A及びJOB－Bが実行されることを前提条件としているが，この場合は，実行されない。

【0011】図1（c）には，図1（a）及び図1（b）に示されたJOB－A，JOB－B，及びJOB－Cの一例として，斜線で示されるJOB－Bの詳細処理フローが示されている。まず，JOB－Bは，ステップS2－0でスタートし，ステップS2－1で実行前のチェックが行われ，続いて，ステップS2－2で，JOB－Bが実際に実行される。JOBの実行後，ステップS2－3で実行後のチェックが行われ，ステップS2－Eで，JOB－Bが終了する。同様な処理フローによって，他のJOB－A，JOB－Cも実行される。

【0012】図2（a）に示されたJOB－A，JOB－B，及びJOB－CのJOBスケジュールレコードには，実行予定日，JOB－A，JOB－B，JOB－Cが実行されるか否かが記録されている。

【0013】また，図2（b）に示されたJOBチェックレコードには，チェックの結果が記録されており，黒星印は，「正常」，「異常」，及び「不実行」の内のいずれかを示している。図1（c）の実行前のチェックには，このJOBスケジュールレコードとJOBチェックレコードが参照される。

【0014】図2（c）に示されたJOB結果更新レコードは，JOB識別（JOB－ID）とチェックレコード逆引テーブルから構成されている。

【0015】図2（d）に示されたJOB実行管理ファイルには，図1（c）のステップS2－1，及びS2－3における各チェックの際に，参照・更新される上記3種類の実行管理ファイル，即ち，JOBスケジュールレコード，JOBチェックレコード，及びJOB結果更新レコードが格納されている。

【0016】ここで，図2（a）のJOBスケジュールレコードは，担当者によって処理の自動運転実施予定日ごとにあらかじめ登録される。また，図2（b）に示すJOBチェックレコード，および，図2（c）のJOB結果更新レコードはJOBが新規に作成され，処理に組み込まれる際にそれぞれのJOB－IDをKEYとして作成される。

【0017】図1（c）に戻ると，斜線で示される実行前チェックには，図2（a）の斜線で示されるJOBスケジュールレコードのレコード内容11，図2（b）の斜線で示されるJOBチェックレコード（のレコード内容）12，13がそれぞれ参照，更新される。また，図2（c）の斜線に示されるJOB結果更新レコード（のレコード内容）14も参照，更新される。

【0018】図3は図1（c）のJOB実行前チェックの詳細な処理の流れを示す図であり，図4は図1（c）のJOB実行後チェックの詳細な処理の流れを示す図である。

【0019】次に動作について説明する。図1（c）の処理フローに示される処理が，図1（a）のジョブストリームによって起動されると最初に，ステップS2－1の実行前チェックプログラムが実行される。その実行前チェックプログラムにおいては，まず，該当する処理予定日をKEYとして図2（a）の斜線で示されるJOBスケジュールレコードが読み込まれる（ステップS2－11）。次に，ステップS2－12において，テーブルサーチによって処理予定のJOB（ここでは，斜線で示される"B"）の実行予定のチェックを行う。ここで，不実行（N）ならば，図4のフローチャートのステップS2－31まで進む。一方，実行（Y）である場合，該当するJOB（ここでは"B"）をKEYとして，図2（b）に示されるJOBチェックレコードを読み込む（ステップS2－13）。次に，斜線で示されるような前提条件JOBテーブルを順次参照することによって，前提条件となっているJOBが全て正常であるかどうかを判定する（ステップS2－14）。その結果，全て正常（Y）である場合は，JOB－Bを実行する（ステップS2－15）。

【0020】JOB－B完了後，図4に示す実行後チェックプログラムが起動される。そのプログラムにおいては，該当JOB（ここでは"B"）をKEYとして，図

5

2（c）の斜線で示されるJOB結果更新レコード14を読み込み（ステップS2－31），チェックレコード逆引テーブルを順次参照する（ステップS2－32，及びS2－36）。逆引テーブルに格納されているJOB（自JOBを前提条件としているJOB：ここでは"C"）をKEYとしたJOBチェックレコード13を読み込み（ステップS2－33），JOB－Bの実行結果のステータス（正常，異常，不実行）を該当テーブルにセットし（ステップS2－34），JOBチェックレコード13の再書込みを行う（ステップS2－35）。チェックレコード逆引テーブルに格納されているすべてのJOBをKEYとした図2（b）のJOBチェックレコードに対し，ステップS2－33～S2－35の処理が完了した時点で，図1（c）に示されたJOB－A，JOB－B，JOB－Cの3の処理を完了する（ステップS2－E）。

【0021】

【発明の効果】以上説明したように，本発明は無人自動運転を含む，複数JOBからなるバッチ処理を行う汎用機システムにおいて，担当員が逐一処理の可否を人為的に判断するのではなく，登録された前提条件JOBをはじめとするJOB属性，および処理実行のスケジュールをもとに自動的に各JOBの実行可否を判定することによって，電算人員の不足しているシステム体制においてもミスのない運用を行うことが可能となるため，作業の正確性向上・作業軽減の実現はもとより，システム異常の場合のリカバリ工数を削減し，ひいては業務自体に支障の少ないシステムサポートが実現可能となる効果がある。

6

【図面の簡単な説明】

【図1】（a）は本発明の実施例における構成と流れを示す説明図である。（b）は（a）の例における各JOBの実行スケジュールと前提条件JOBを示す図である。（c）は（a）のJOB－Bの処理フローを示す図である。

【図2】（a）はJOBスケジュールレコードを示す図である。（b）はJOBチェックレコードを示す図である。（c）はJOB結果更新レコードを示す図である。（d）はJOB管理ファイルを示す図である。

【図3】図1（c）のJOB実行前チェックの詳細な処理の流れを示す図である。

【図4】図1（c）のJOB実行後チェックの詳細な処理の流れを示す図である。

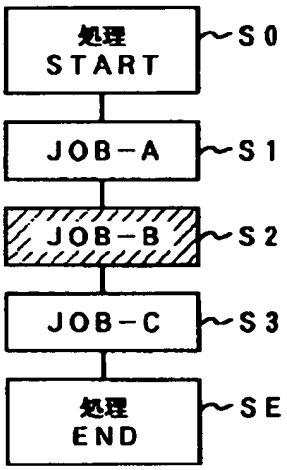
【符号の説明】

10　JOB実行管理ファイル
11　JOBスケジュールレコード
12，13　JOBチェックレコード
14　JOB結果更新レコード

【図1】

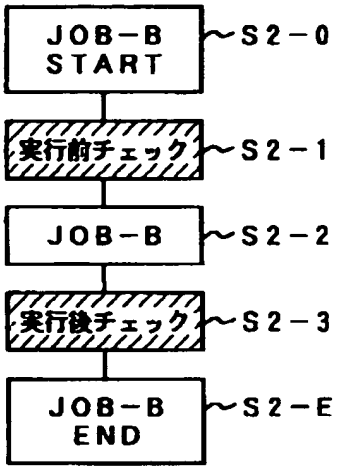
(a)

バッチ処理
ジョブストリーム



(c)

JOB－Bの処理フロー



(b)

各JOBの実行スケジュールと前提条件JOB

JOB	スケジュール	前提条件JOB
A	実　行	－
B	実　行	A
C	不実行	A，B

【図2】

(a)

《JOBスケジュールレコード》

SCH	日付	A	B	C	…	
SCH	予定日	実行	実行	不実行	…	～11

(b)

《JOBチェックレコード》★は"正常"or"異常"or"不実行"

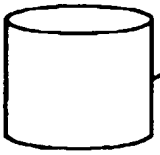
CHK	JOB－ID	前提条件JOBテーブル				
CHK	A	－	－	－	…	
CHK	B	A：★	－	－	…	～12
CHK	C	A：★	B：★	－	…	～13

(c)

《JOB結果更新レコード》

END	JOB－ID	チェックレコード逆引テーブル				
END	A	B	C	－	…	
END	B	C	－	－	…	～14

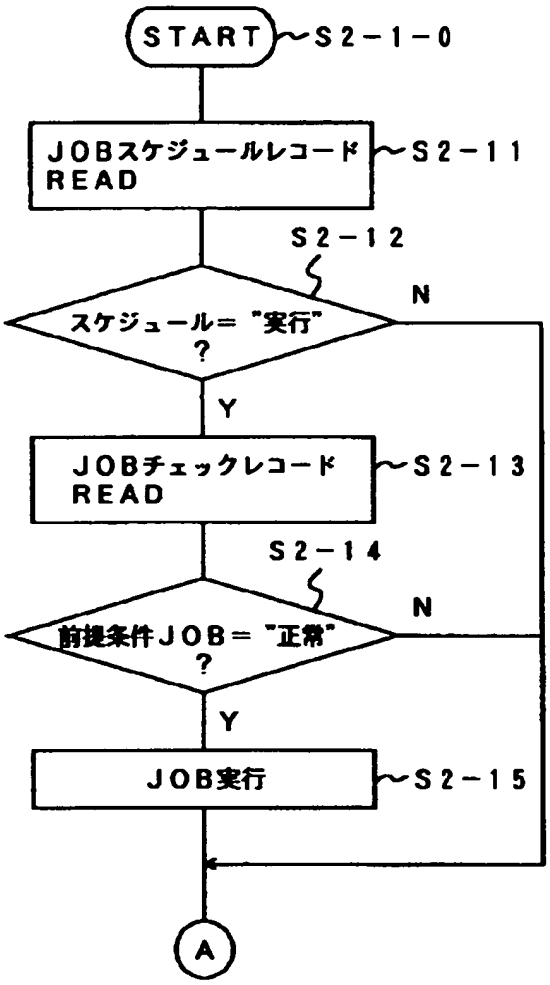
(d)



～10　JOB実行管理ファイル

【図3】

《JOB実行前チェック》



【図4】

《JOB実行後チェック》